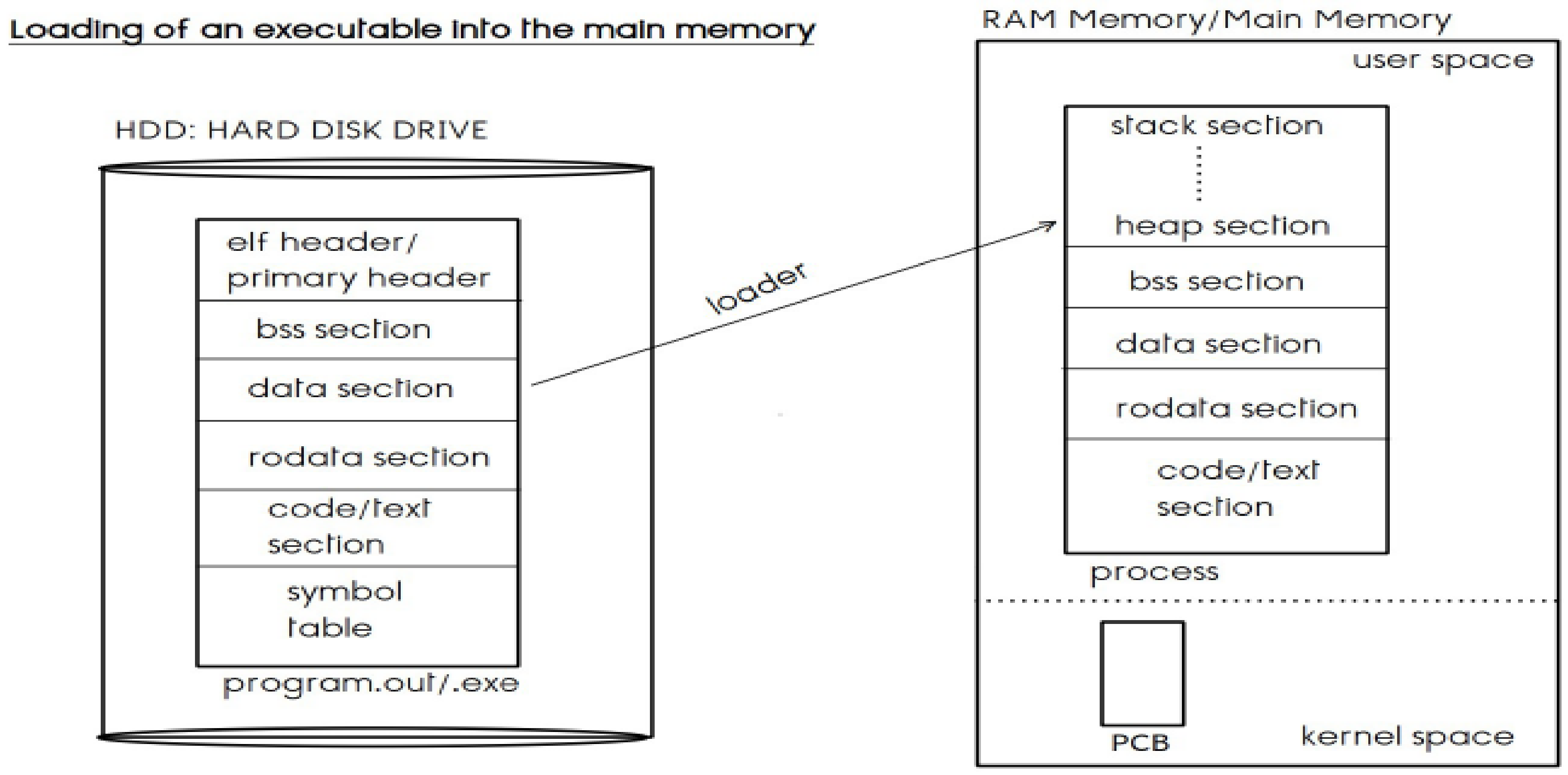
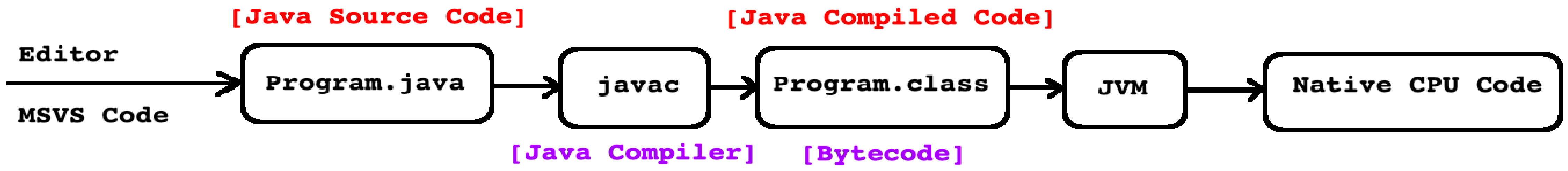
Core Java
JVM (Java Virtual Machine) Architecture

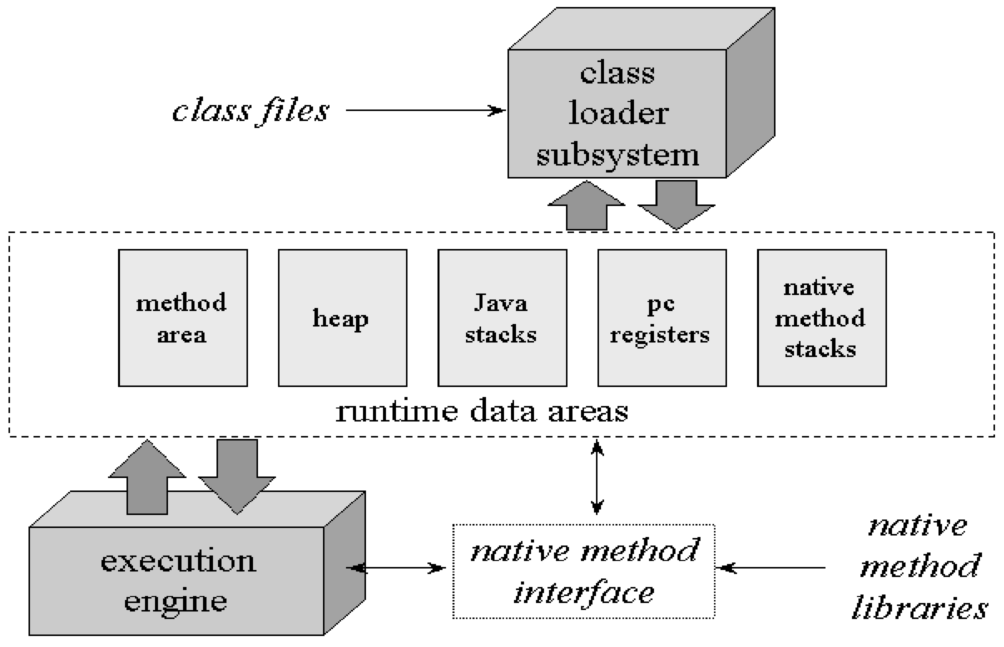
Dr.Akshita S.Chanchlani

Process Loaded in Main Memory



JVM Architechture





[JVM Architecture]

CLASS LOADER SUB SYSTEM

JVM MEMORY AREAS

EXECUTION ENGINE

Components of JVM : 1. Class Loader Sub System (Loading , Linking , Initialization)

1. **Class Loader Sub System**

<u>LOADING</u>

- Bootstrap class loader (rt.jar) : Load Java builtin classes from jre/lib jars.
- Extended class Loader (jre) : Load extended classes from jre/lib/ext directory
- Application class Loader (class path) :Load classes from the application classpath

Reads the class from the disk and loads into JVM method (memory) area.

<u>LINKING</u>

- Verification , Preparation , Resolution
- Verification: Byte code verifier does verification process. Ensure that class is compiled by a valid compiler and not tampered.
- Preparation: Memory is allocated for static members and initialized with their default values.
- Resolution: Symbolic references in constant pool are replaced by the direct references

<u>INTIALIZATION</u>

- Static variables of the class are assigned with assigned values
- Execute the static blocks if present

Components of JVM : 2.Runtime Data Areas/ JVM Memory Areas

- While execution, memory is required for byte code, objects, variables, etc.
- There are five areas: Method area, Heap area, Stack area, PC Registers, Native Method Stack area.

Method Area:

- Created at while JVM startup, Shared by all threads (global).
- Class contents (for all classes) are loaded into Method area.
- Method area also holds constant pool for all loaded classes.

Heap Area :

- Created at while JVM startup , Shared by all threads (global).
- All allocated objects (with new keyword) are stored in heap.
- The class Metadata is stored in a java.lang.Class object (in heap) once class is loaded

Stack Area

- Separate stack is created for each thread in JVM (while creating thread).
- When a method is called from the stack, a new FAR (stack frame) is created on its stack.
- This stack frame contains local variable array, operand stack, and other frame data, When method returns, the stack frame is destroyed.

PC Registers

- Separate PC register is created for each thread. It maintains address of the next instruction executed by the thread.
- After an instruction is completed, the address in PC is auto-incremented.

Native method stack area

- Separate native method stack is created for each thread in JVM (while creating thread).
- When a native method is called from the stack, a stack frame is created on its stack.

Components of JVM : 3. Execution Engine

3. Execution Engine

Interpreter

- Convert byte code into machine code and execute it (instruction by instruction).
- Each method is interpreted by the interpreter at least once.
- If method is called frequently, interpreting it each time slow down the execution of the program. This limitation is overcomed by JIT (added in Java 1.1).

JIT compiler

JIT stands for Just In Time compiler.
Primary purpose of the JIT compiler to improve the performance.
If a method is getting invoked multiple times, the JIT compiler convert it into native code and cache it.
If the method is called next time, its cached native code is used to speedup execution process.

Profiler

Tracks resource (memory, threads, ...) utilization for execution.
Part of JIT that identifies hotspots. It counts number of times any method is executing. If the number is more than a threshold value, it is considered as hotspot.

Garbage Collector

When any object is unreferenced, the GC release its memory.

JNI (Java Native Interface) : JNI acts as a bridge between Java method calls and native method implementations.

SUNBEAM

Thank You.

[akshita.chanchlani@sunbeaminfo.com]